Car-Price-Prediction-Project

Submitted by:

Dolypona Das

ACKNOWLEDGMENT

I would like to thank our SME (Shubham Yadav) for his expert advice and encouragement throughout this project.

INTRODUCTION

- ## Business Problem Framing

In this project,we have to make carprice valuation model using new machine learning models from new data. Because with the change in market due to covid 19 impact, our client is facing problems with their previous car price valuation machine learning models.

- ## Conceptual Background of the Domain Problem

1.Firstly,we will prepare our own dataset using web scraping.
2. After that we will check whether the project is a regression type or a classification type.
3. We will also check whether our dataset is balanced or imbalanced. If it is an imbalanced one, we will apply sampling techniques to balance the dataset.
4. Then we will do model building and check its accuracy.
5. Our main motto is to build a model with good accuracy and for that we will also go for hyperparameter tuning.

- ## Review of Literature

I am summarizing my research done on the topic.
▪ I have created my own dataset using web scraping and imported important libraries for my project.
▪ I have created the dataframe.
▪ I have analysed my data by checking its shape, number of columns, presence of null values if any and checking the datatypes.
▪ Then I have done some data cleaning steps,e.g Checking the value counts of the target variable, dropping some irrelevant columns from the dataset, checking correlation between the dependant and independent variables using heatmap, visualizing data using distribution plots, detecting and removing skewness in my data if any, outliers detection using boxplots and removing them, balancing dataset using

randomoversampler method, splitting the data into independent and dependant variables and finally scaling the data.

Then I have used 5 regressor models ,out of which AdaBoostRegressor is giving a good accuracy score of 99% after hyperparameter tuning.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

If you look at data science, we are actually using mathematical models to model (and hopefully through the model to explain some of the things that we have seen) business circumstances, environment etc and through these model, we can get more insights such as the outcomes of our decision undertaken, what should we do next or how shall we do it to improve the odds. So mathematical models are important, selecting the right one to answer the business question can bring tremendous value to the organization.
Here I am using AdaBoostRegressor with accuracy 99% after hyper parameter tuning.

- Data Sources and their formats

Data Source: The read_csv function of the pandas library is used to read the content of a CSV file into the python environment as a pandas DataFrame. The function can read the files from the OS by using proper path to the file.
Data description: Pandas describe() is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values.

- Data Preprocessing Done

  - I have checked for null values and there are some null values present,I have removed it using SimpleImputer method.
  - I have label encoded the object type columns in the dataset.

- I have checked the correlation between dependant and independent variables using heatmap. I have seen most of the independent variables are correlated with each other and the target variable is positively correlated with a very few independent variables.
- I have done some visualization using histogram**.**
- I have checked outliers using boxplots ,but no outliers are present.
- I also have checked for skewness in my data, but the skewness present is very negligible, so I don't consider it.
- I have splitted the dependant and independent variables into x and y.
- I have scaled the data using StandardScaler method and made my data ready for model building.

- Hardware and Software Requirements and Tools Used
  - Hardware requirements:
    Processor: Intel(R) Celeron(R) CPU N3050 @1.60 GHz 1.60 GHz
    RAM: 3.92 GB
    System type: 64-bit operating system, x64-based processor
  - Software requirements:
    Python: One of the most used programming languages
    Tools used:
    Jupyter notebook: Jupyter is a free, open-source, interactive web tool known as a computational notebook where I have written my python codes.
    NumPy: NumPy is an open-source numerical Python library. NumPy contains a multi-dimensional array and matrix data structures.
    Pandas: Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.
    Matplotlib: It is a Python library used for plotting graphs with the help of other libraries like Numpy and Pandas. It is a powerful tool for visualizing data in Python.
    Seaborn: It is also a Python library used for plotting graphs with the help of Matplotlib, Pandas, and Numpy.

Scikit-learn: It is probably the most useful library for machine learning in Python.The sklearn library contains a lot of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction.
Scipy.stats: This module contains a large number of probability distributions as well as a growing library of statistical functions.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

1.To check the correlation among the data,I have used **heatmap** to visualize it.
2. To get a clear view of the columns visually,I have used **distribution plots**.
3. For checking outliers,I have used **boxplots.**

4. For scaling the data, I have used **StandardScaler** method.
5. For training and testing the data, I have imported **train_test_split library** from scikit-learn.

6. For model building, I have used 5 Regressor models(DecisionTreeRegressor(),KNeighborsRegressor(),AdaBoostRegressor(),LinearRegression(),GradientBoostingRegressor()),out of which **AdaBoostRegressor** model is the best model for my dataset.

7. For better accuracy of the model, I have used **hyperparameter tuning (RandomizedSearchCV).**

- Testing of Identified Approaches (Algorithms)

I have used 5 algorithms for testing.

- **DecisionTreeRegressor**- Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller

and smaller subsets while at the same time an associated decision tree is incrementally developed.

- **KNeighborsRegressor**- In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbors.
- **AdaBoostRegressor**- An AdaBoost regressor is a meta-estimator that begins by fitting a regressor on the original dataset and then fits additional copies of the regressor on the same dataset but where the weights of instances are adjusted according to the error of the current prediction.
- **LinearRegression**- Linear regression is a linear approach for modelling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables).
- **GradientBoostingRegressor**- Gradient boosting algorithm is one of the most powerful algorithms in the field of machine learning.Gradient boosting algorithm can be used for predicting not only continuous target variable (as a Regressor) but also categorical target variable (as a Classifier).

- Run and Evaluate selected models

AdaBoostRegressor:

```
In [27]: #Model Building(Finding the best random state)

model=[DecisionTreeRegressor(),KNeighborsRegressor(),AdaBoostRegressor(),LinearRegression(),GradientBoostingRegressor()]
max_r2_score=0
for r_state in range(40,90):
    train_x,test_x,train_y,test_y=train_test_split(x,y,random_state=r_state,test_size=0.33)
    for i in model:
        i.fit(train_x,train_y)
        pre=i.predict(test_x)
        r2_sc=r2_score(test_y,pre)
        print('r2 score correspond to random state',r_state,'is',r2_sc)
        if r2_sc>max_r2_score:
            max_r2_score=r2_sc
            final_state=r_state
            final_model=i
print()
print()
print()
print()
print('max r2 score correspond to random state',final_state,'is',max_r2_score,'and model is',final_model)
```

```
r2 score correspond to random state 87 is -0.02345468206870538
r2 score correspond to random state 87 is 0.20719921668305274
r2 score correspond to random state 87 is -0.07266077192139941
r2 score correspond to random state 87 is -0.2817027114462134
r2 score correspond to random state 88 is -0.5283642362586722
r2 score correspond to random state 88 is -0.17815145239178598
r2 score correspond to random state 88 is -0.6363761741452101
r2 score correspond to random state 88 is -0.17146405918640006
r2 score correspond to random state 88 is -0.562329432407539
r2 score correspond to random state 89 is -81.90134166699453
r2 score correspond to random state 89 is -75.10340668523762
r2 score correspond to random state 89 is -83.74977559052957
r2 score correspond to random state 89 is -47.308304906426244
r2 score correspond to random state 89 is -37.359458335983255



max r2 score correspond to random state 60 is 0.9787986716414042 and model is AdaBoostRegressor()
```

Above I am using for loop which helps me to provide the r2 score at each random state and for the best state where r2 score is maximum has come as output value.

The best model is AdaBoostRegressor and the r2 score is 98% to random state 60.

```
In [39]: #Hyperparameter tuning using RandomizedSearchCV
from sklearn.model_selection import RandomizedSearchCV

In [50]: #List of parameters to pass
n_estimators = [10,50,100]
loss=['linear','square','exponential']
#max_features = ['auto', 'sqrt']
#max_depth = [2, 3, 5]
#min_samples_split = [2, 4, 6]
#min_samples_leaf = [1, 2, 4, 6]
learning_rate=[0.1]

In [51]: #Creating random grid

random_grid = {'n_estimators': n_estimators,
               'loss':loss,
               'learning_rate':learning_rate}

In [52]: # Random search of parameters, using 5 fold cross validation,
# search across 100 different combinations
ab_random = RandomizedSearchCV(estimator = ab, param_distributions = random_grid,scoring='neg_mean_squared_error', n_iter = 10, c
```

```
Out[53]: RandomizedSearchCV(cv=5, estimator=AdaBoostRegressor(), n_jobs=1,
                            param_distributions={'learning_rate': [0.1],
                                                 'loss': ['linear', 'square',
                                                          'exponential'],
                                                 'n_estimators': [10, 50, 100]},
                            random_state=60, scoring='neg_mean_squared_error',
                            verbose=2)

In [54]: ab_random.best_params_

Out[54]: {'n_estimators': 50, 'loss': 'exponential', 'learning_rate': 0.1}

In [55]: ab=AdaBoostRegressor(n_estimators=50,loss='exponential',learning_rate=0.1)
         ab.fit(train_x,train_y)
         ab.score(train_x,train_y)
         pred=ab.predict(test_x)
         abr=r2_score(test_y,pred)
         print('R2 score:',abr*100)

         R2 score: 98.91148020928856
```

After hyperparameter tuning ,the accuracy is 99%.

- Key Metrics for success in solving problem under consideration

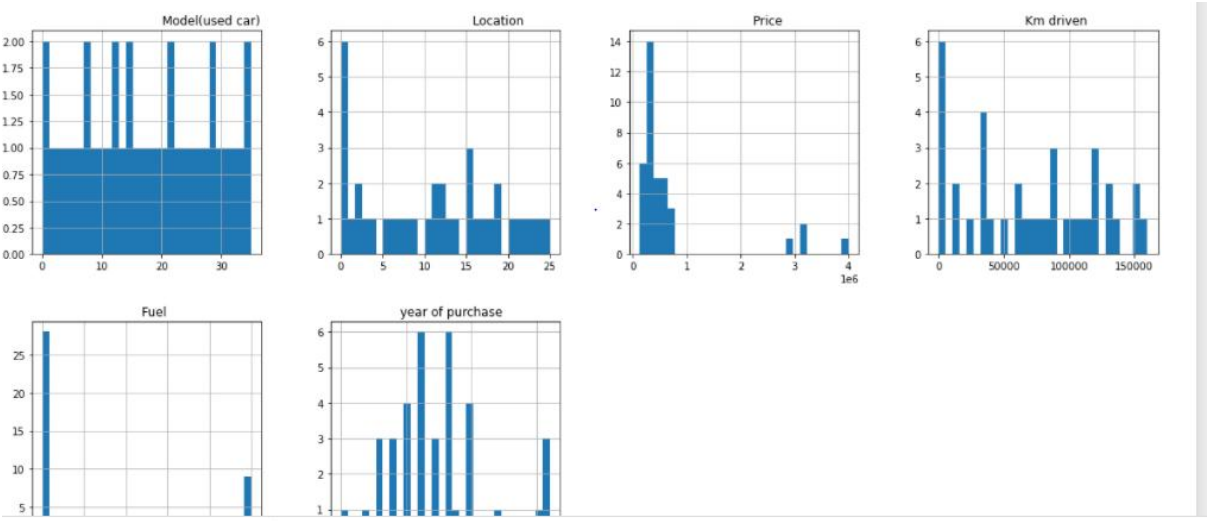
```
In [38]: from sklearn.metrics import mean_absolute_error
         print('Mean Absolute Error:',mean_absolute_error(test_y,y_pred))
         print('Mean Squared Error:',mean_squared_error(test_y,y_pred))
         print('Root Mean Absolute Error:',np.sqrt(mean_absolute_error(test_y,y_pred)))

         Mean Absolute Error: 95321.63992673994
         Mean Squared Error: 15556351773.986591
         Root Mean Absolute Error: 308.7420281185248
```

- Visualizations

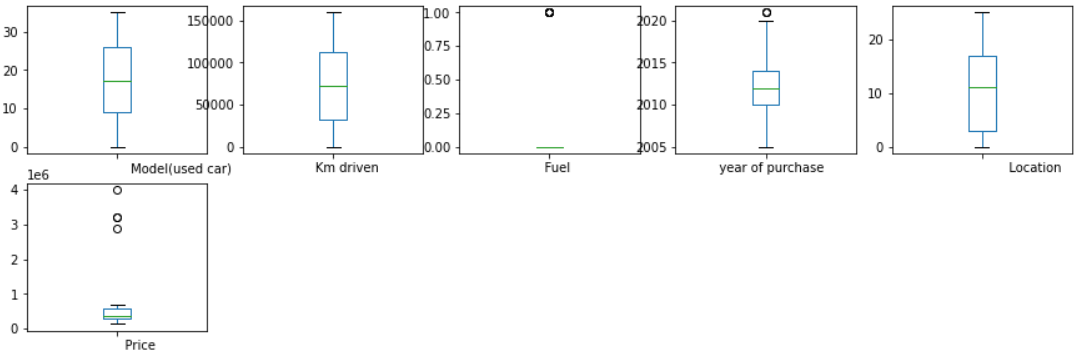
Correlation matrix using heatmap-checking correlation between dependant and independent variables.

```
In [21]: #Visualizing the correlation
         plt.figure(figsize=(14,14))
         sns.heatmap(df.corr(), annot=True, fmt='.0%')
```

Histogram-



Boxplots for outlier detection-



- Interpretation of the Results

In the visualization part, I have seen how my data looks like using heatmap, boxplot, distribution plots, histogram etc.

In the pre-processing part, I have cleaned my data using many methods like SimpleImputer,LabelEncoder etc.

In the modelling part, I have designed our model using algorithm like AdaBoostRegressor.
The accuracy , Mean Absolute Error, Mean Squared Error, Root Mean Absolute Error are achieved for the model.

CONCLUSION

- Key Findings and Conclusions of the Study

The key findings are we have to study the data very clearly so that we are able to decide which data are relevant for our findings. The techniques that I have used are heatmap, SimpleImputer,LabelEncoder etc.
The conclusion of our study is we have to achieve a model with good accuracy and f1-score.

- Learning Outcomes of the Study in respect of Data Science

We will develop relevant programming abilities. We will demonstrate proficiency with statistical analysis of data. We will develop the ability to build and assess data-based models. We will execute statistical analysis with professional statistical software. The best algorithm for this project according to my work is

AdaBoostRegressor because the accuracy that I have achieved is quite satisfactory than the other model.

- ## Limitations of this work and Scope for Future Work

The scope for future work is to collect as many data as we can so that the model can be built more efficiently.